

windmill-labs / windmill Public

<> Code Issues 533 Pull requests 169 Discussions Actions Projects

Commit 942fb62



rubenfiszel committed on Jan 11 · ✖ 3 / 22

nit tightening

main · v1.687.0 ... pr-assets

1 parent [f0fd1c5](#) commit 942fb62

4 files changed +167 -74 lines changed

Top



backend/windmill-api/src

- folders.rs
- jobs.rs
- resources.rs
- settings.rs

4 files changed +167 -74 lines changed



backend/windmill-api/src/folders.rs



```

@@ -130,6 +130,18 @@ async fn list_foldernames(
130 130     Ok(Json(rows))
131 131 }
132 132
133 + fn validate_owner(owner: &str) -> Result<()> {
134 +     if !owner
135 +         .chars()
136 +         .all(|c| c.is_ascii_alphanumeric() || c == '_' || c == '/' || c == '-')
137 +     {

```

```

138 +         return Err(error::Error::BadRequest(
139 +             "Invalid owner: must contain only alphanumeric characters,
           underscores, hyphens, or slashes".to_string(),
140 +         ));
141 +     }
142 +     Ok(())
143 + }
144 +
133 145     async fn check_name_conflict<'c>(
134 146         tx: &mut Transaction<'c, Postgres>,
135 147         w_id: &str,
           ↓
           ↑
202 214         ));
203 215     }
204 216
205 -     if let Err(e) =
217 +     if let Err(e) =
206 218         sqlx::query_as!(
207 219             Folder,
208 220             "INSERT INTO folder (workspace_id, name, display_name, owners,
           extra_perms, summary, created_by, edited_at) VALUES ($1, $2, $3, $4, $5, $6,
           $7, now())",
           ↓
           ↑
262 274         )
263 275         .await?;
264 276
265 -     log_folder_permission_change(
266 -         &mut *tx,
267 -         &w_id,
268 -         &ng.name,
269 -         &authed.username,
270 -         "create",
271 -         None,
272 -     )
273 -     .await?;
277 +     log_folder_permission_change(&mut *tx, &w_id, &ng.name, &authed.username,
           "create", None)
278 +     .await?;

```

274 279

275 280 tx.commit().await?;

276 281

@@ -381,7 +386,8 @@ async fn update_folder(

381 386 if let Some(extra_perms) = ng.extra_perms {

382 387 if !extra_perms.is_object() {

383 388 return Err(windmill_common::error::Error::BadRequest(format!(

384 - "extra_perms must be an object, received {}",

extra_perms.to_string()

389 + "extra_perms must be an object, received {}",

390 + extra_perms.to_string()

385 391));

386 392 }

387 393 sqlb.set(
@@ -447,15 +453,8 @@ async fn update_folder(

447 453 .await?;

448 454 }

449 455 if extra_perms_changed {

450 - log_folder_permission_change(

451 - &mut *tx,

452 - &w_id,

453 - &name,

454 - &authed.username,

455 - "update_acl",

456 - None,

457 -)

458 - .await?;

456 + log_folder_permission_change(&mut *tx, &w_id, &name, &authed.username,
"update_acl", None)

457 + .await?;

459 458 }

460 459

461 460 tx.commit().await?;

@@ -695,6 +694,7 @@ async fn add_owner(

695 694 .fetch_optional(&mut *tx)

696 695 .await?;

697 696

697	+	<code>validate_owner(&owner)?;</code>
698	698	<code>sqlx::query(&format!(</code>
699	699	<code> "UPDATE folder SET extra_perms = jsonb_set(extra_perms, '{{\</code>
		<code>{owner}\\"}}', to_jsonb(\$1), \</code>
700	700	<code> true) WHERE name = \$2 AND workspace_id = \$3 RETURNING extra_perms"</code>
		<code>@@ -747,6 +747,7 @@ async fn remove_owner(</code>
747	747	
748	748	<code> not_found_if_none(get_folderopt(&mut tx, &w_id, &name).await?, "Folder",</code>
		<code> &name)?;</code>
749	749	<code> require_is_owner(&authed, &name)?;</code>
750	+	<code>validate_owner(&owner)?;</code>
750	751	
751	752	<code> let folder = sqlx::query!(</code>
752	753	<code> "UPDATE folder SET owners = array_remove(owners, \$1::varchar) WHERE</code>
		<code>name = \$2 AND workspace_id = \$3 AND \$1 = ANY(owners) RETURNING name",</code>
		<code>@@ -758,7 +759,10 @@ async fn remove_owner(</code>
758	759	<code> .await?;</code>
759	760	
760	761	<code> if folder.is_none() && write.is_none() {</code>
761	-	<code> return Ok(format!("Owner {} is already not a member of folder {}",</code>
		<code>owner, name));</code>
762	+	<code> return Ok(format!(</code>
763	+	<code> "Owner {} is already not a member of folder {}",</code>
764	+	<code> owner, name</code>
765	+	<code>));</code>
762	766	<code> }</code>
763	767	
764	768	<code> if let Some(write) = write {</code>
		<code>@@ -775,11 +779,13 @@ async fn remove_owner(</code>
775	779	<code> .flatten();</code>
776	780	
777	781	<code> if folder.is_none() && old_write.is_none_or(ow ow == write) {</code>
778	-	<code> return Ok(format!("Owner {} is already not a member of folder {}</code>
		<code>and write permission was already {}", owner, name, write));</code>
782	+	<code> return Ok(format!(</code>
783	+	<code> "Owner {} is already not a member of folder {} and write</code>
		<code>permission was already {}",</code>
784	+	<code> owner, name, write</code>
785	+	<code>));</code>

```

779 786      }
780 787    }
781 788
782 -
783 789      audit_log(
784 790          &mut *tx,
785 791          &authed,
@@ -796,7 +802,15 @@ async fn remove_owner(
796 802          Some(false) => "grant_viewer_only",
797 803          None => "revoke_all",
798 804      };
799 -      log_folder_permission_change(&mut *tx, &w_id, &name, &authed.username,
change_type, Some(&owner)).await?;
805 +      log_folder_permission_change(
806 +          &mut *tx,
807 +          &w_id,
808 +          &name,
809 +          &authed.username,
810 +          change_type,
811 +          Some(&owner),
812 +      )
813 +      .await?;
800 814
801 815      tx.commit().await?;
802 816

```

▼ backend/windmill-api/src/jobs.rs

```

@@ -440,24 +440,24 @@ async fn get_flow_env_by_flow_job_id(
440 440    ) -> windmill_common::error::JsonResult<Box<JsonRawValue>> {
441 441      let flow_env = sqlx::query_scalar!(
442 442          r#"
443 -          SELECT
444 -          CASE
445 +          SELECT
446 +          CASE
447 447          WHEN flow_version.id IS NOT NULL THEN
448 448              (flow_version.value -> 'flow_env' -> $3) #> $4
          ELSE
              (root_job.raw_flow -> 'flow_env' -> $3) #> $4

```

```

449 449          END AS "flow_env: sqlx::types::Json<Box<RawValue>>"
450 -          FROM
451 +          FROM
451 451          v2_job current_job
452 -          JOIN
453 +          JOIN
453 453          v2_job root_job ON root_job.id =
          COALESCE(current_job.root_job, current_job.flow_innermost_root_job,
          current_job.parent_job, current_job.id)
454 454          AND root_job.workspace_id = current_job.workspace_id
455 455          LEFT JOIN
456 456          flow_version ON flow_version.id = root_job.runnable_id
457 457          AND flow_version.path = root_job.runnable_path
458 458          AND flow_version.workspace_id = root_job.workspace_id
459 -          WHERE
460 -          current_job.id = $1 AND
461 +          WHERE
462 +          current_job.id = $1 AND
          current_job.workspace_id = $2"#,
461 461          flow_job_id,
462 462          w_id,
463 463
@@ -4341,12 +4341,12 @@ pub async fn run_flow_by_version_inner(
4341 4341
4342 4342          let flow_path = sqlx::query_scalar!(
4343 4343          r#"
4344 -          SELECT
4345 -          path
4346 -          FROM
4347 -          flow_version
4348 -          WHERE
4349 -          id = $1 AND
4350 +          SELECT
4351 +          path
4352 +          FROM
4353 +          flow_version
4354 +          WHERE
4355 +          id = $1 AND
          workspace_id = $2
4351 4351          "#,

```

4352	4352	version,
		@@ -4953,10 +4953,10 @@ pub async fn run_wait_result_internal(
4953	4953	result AS \"result:
		sqlx::types::Json<Box<RawValue>>\",
4954	4954	result_columns,
4955	4955	status = 'success' AS \"success!\"
4956	-	FROM
4956	+	FROM
4957	4957	v2_job_completed
4958	-	WHERE
4959	-	id = \$1 AND
4958	+	WHERE
4959	+	id = \$1 AND
4960	4960	workspace_id = \$2
4961	4961	\",
4962	4962	uuid,
		@@ -5960,12 +5960,12 @@ pub async fn run_wait_result_flow_by_version(
5960	5960	
5961	5961	let flow_path = sqlx::query_scalar!(
5962	5962	r#"
5963	-	SELECT
5964	-	path
5965	-	FROM
5966	-	flow_version
5967	-	WHERE
5968	-	id = \$1 AND
5963	+	SELECT
5964	+	path
5965	+	FROM
5966	+	flow_version
5967	+	WHERE
5968	+	id = \$1 AND
5969	5969	workspace_id = \$2
5970	5970	"#,
5971	5971	version,
		@@ -6925,12 +6925,12 @@ async fn run_dynamic_select(
6925	6925	None => {

```

6926 6926         let dynamic_input = sqlx::query_scalar!(
6927 6927             r#"
6928 -             SELECT
6929 -                 schema
6930 -             FROM
6931 -                 flow
6932 -             WHERE
6933 -                 workspace_id = $1 AND
6928 +             SELECT
6929 +                 schema
6930 +             FROM
6931 +                 flow
6932 +             WHERE
6933 +                 workspace_id = $1 AND
6934 6934                 path = $2
6935 6935             "#,
6936 6936             &w_id,
        @@ -7235,6 +7235,9 @@ impl Hash for JobUpdate {
        }
7235 7235     }
7236 7236
7237 7237     async fn get_log_file(Path((_w_id, file_p)): Path<(String, String)>) ->
        error::Result<Response> {
7238 +         if file_p.contains("..") {
7239 +             return Err(error::Error::BadRequest("Invalid path".to_string()));
7240 +         }
7238 7241         let local_file = format!("{TMP_DIR}/logs/{file_p}");
7239 7242         if tokio::fs::metadata(&local_file).await.is_ok() {
7240 7243             let mut file =
                tokio::fs::File::open(local_file).await.map_err(to_anyhow)?;
        @@ -7647,9 +7650,9 @@ async fn get_flow_stream_delta(
7647 7650         if let Some(job_id) = flow_stream_job_id {
7648 7651             let record = sqlx::query!(
7649 7652                 "
7650 -                 SELECT
7651 -                 string_agg(stream, '' order by idx asc) as stream,
7652 -                 max(idx) + 1 as offset
7653 +                 SELECT
7654 +                 string_agg(stream, '' order by idx asc) as stream,

```

7655	+	max(idx) + 1 as offset
7653	7656	FROM job_result_stream_v2
7654	7657	WHERE job_id = \$2 AND idx >= \$1
7655	7658	",
⋮ ↓ ↑ ⋮		@@ -7710,15 +7713,15 @@ async fn get_job_update_data(
7710	7713	let r = sqlx::query!(
7711	7714	"
7712	7715	WITH result_stream AS (
7713	-	SELECT
7714	-	string_agg(stream, '' order by idx asc) as
		stream,
7715	-	job_id,
7716	-	max(idx) + 1 as offset
7716	+	SELECT
7717	+	string_agg(stream, '' order by idx asc) as
		stream,
7718	+	job_id,
7719	+	max(idx) + 1 as offset
7717	7720	FROM job_result_stream_v2
7718	7721	WHERE job_id = \$2 AND idx >= \$3
7719	7722	GROUP BY job_id
7720	7723)
7721	-	SELECT
7724	+	SELECT
7722	7725	jc.result as \"result:
		sqlx::types::Json<Box<RawValue>>\",
7723	7726	v2_job.tag,
7724	7727	v2_job_queue.running as \"running: Option<bool>\",
⋮ ↓ ↑ ⋮		@@ -7760,10 +7763,10 @@ async fn get_job_update_data(
7760	7763	let r = sqlx::query!(
7761	7764	"
7762	7765	WITH result_stream AS (
7763	-	SELECT
7764	-	string_agg(stream, '' order by idx asc) as
		stream,
7765	-	job_id,
7766	-	max(idx) + 1 as offset
7766	+	SELECT

7767	+	string_agg(stream, '' order by idx asc) as
		stream,
7768	+	job_id,
7769	+	max(idx) + 1 as offset
7767	7770	FROM job_result_stream_v2
7768	7771	WHERE job_id = \$1 AND idx >= \$3
7769	7772	GROUP BY job_id
⌵		@@ -7803,10 +7806,10 @@ async fn get_job_update_data(⌶
7803	7806	let q = sqlx::query!(
7804	7807	"
7805	7808	WITH result_stream AS (
7806	-	SELECT
7807	-	string_agg(stream, '' order by idx asc) as
		stream,
7808	-	job_id,
7809	-	max(idx) + 1 as offset
7809	+	SELECT
7810	+	string_agg(stream, '' order by idx asc) as
		stream,
7811	+	job_id,
7812	+	max(idx) + 1 as offset
7810	7813	FROM job_result_stream_v2
7811	7814	WHERE job_id = \$2 AND idx >= \$3
7812	7815	GROUP BY job_id
⌵		@@ -7874,10 +7877,10 @@ async fn get_job_update_data(⌶
7874	7877	let mut record = sqlx::query!(
7875	7878	"
7876	7879	WITH result_stream AS (
7877	-	SELECT
7878	-	string_agg(stream, '' order by idx asc) as stream,
7879	-	job_id,
7880	-	max(idx) + 1 as offset
7880	+	SELECT
7881	+	string_agg(stream, '' order by idx asc) as stream,
7882	+	job_id,
7883	+	max(idx) + 1 as offset
7881	7884	FROM job_result_stream_v2
7882	7885	WHERE job_id = \$3 AND idx >= \$8

7883 7886

GROUP BY job_id



backend/windmill-api/src/resources.rs



@@ -1469,6 +1469,55 @@ struct GitRepositoryResource {

1469 1469 branch: Option<String>,

1470 1470 }

1471 1471

```
1472 + /// Validates a git URL to prevent git option injection attacks.
1473 + /// Git URLs starting with '-' could be interpreted as command-line options.
1474 + fn validate_git_url(url: &str) -> Result<()> {
1475 +     let url = url.trim();
1476 +     if url.is_empty() {
1477 +         return Err(Error::BadRequest("Git URL cannot be empty".to_string()));
1478 +     }
1479 +     if url.starts_with('-') {
1480 +         return Err(Error::BadRequest(
1481 +             "Git URL cannot start with '-' (potential option
1482 +             injection)".to_string(),
1483 +         ));
1484 +     }
1485 +     // Block other potentially dangerous patterns
1486 +     if url.contains('\0') || url.contains('\n') || url.contains('\r') {
1487 +         return Err(Error::BadRequest(
1488 +             "Git URL contains invalid characters".to_string(),
1489 +         ));
1490 +     }
1491 +     Ok(())
1492 + }
1493 + /// Validates a git branch/ref name to prevent injection attacks.
1494 + fn validate_git_ref(ref_name: &str) -> Result<()> {
1495 +     let ref_name = ref_name.trim();
1496 +     if ref_name.is_empty() {
1497 +         return Err(Error::BadRequest("Git ref cannot be empty".to_string()));
1498 +     }
1499 +     if ref_name.starts_with('-') {
1500 +         return Err(Error::BadRequest(
1501 +             "Git ref cannot start with '-' (potential option
1502 +             injection)".to_string(),
```

```

1502 +     ));
1503 + }
1504 + // Git ref names have specific rules - block dangerous characters
1505 + if ref_name.contains('\0')
1506 +     || ref_name.contains('\n')
1507 +     || ref_name.contains('\r')
1508 +     || ref_name.contains("..")
1509 +     || ref_name.contains("@{")
1510 +     || ref_name.ends_with('.')
1511 +     || ref_name.ends_with('/')
1512 +     || ref_name.contains("//")
1513 + {
1514 +     return Err(Error::BadRequest(
1515 +         "Git ref contains invalid characters or patterns".to_string(),
1516 +     ));
1517 + }
1518 + Ok(())
1519 + }
1520 +

```

```

1472 1521 #[derive(Serialize)]

```

```

1473 1522 struct GitCommitHashResponse {

```

```

1474 1523     commit_hash: String,

```



```

@@ -1629,14 +1678,21 @@ async fn get_repo_latest_commit_hash(

```

```

1629 1678     git_resource: &GitRepositoryResource,

```

```

1630 1679     git_ssh_command: Option<String>,

```

```

1631 1680 ) -> Result<String> {

```

```

1632 -     let mut git_cmd = Command::new("git");

```

```

1681 + // Validate URL and branch to prevent option injection attacks

```

```

1682 + validate_git_url(&git_resource.url)?;

```

```

1633 1683

```

```

1634 1684     let ref_spec = git_resource

```

```

1635 1685         .branch

```

```

1636 1686         .as_deref()

```

```

1637 1687         .filter(|s| !s.is_empty())

```

```

1638 1688         .unwrap_or("HEAD");

```

```

1639 1689

```

```

1690 + // Validate ref_spec if it's not the default HEAD

```

```

1691 + if ref_spec != "HEAD" {

```

```

1692 +     validate_git_ref(ref_spec)?;

```

```

1693 +   }
1694 +
1695 +   let mut git_cmd = Command::new("git");
1640 1696       git_cmd.args(["ls-remote", &git_resource.url, ref_spec]);
1641 1697       if let Some(git_ssh_command) = git_ssh_command {
1642 1698           git_cmd.env("GIT_SSH_COMMAND", git_ssh_command);

```



backend/windmill-api/src/settings.rs

```

@@ -709,11 +709,31 @@ async fn setup_custom_instance_pg_database_inner(
709 709     // Validate name to ensure it only contains alphanumeric characters
710 710     // Prevents SQL injection on the instance database
711 711     lazy_static::lazy_static! {
712 -         static ref VALID_NAME: regex::Regex = regex::Regex::new(r"^[a-zA-Z0-
9_]+$").unwrap();
712 +         // Must start with a letter, then alphanumeric/underscore
713 +         static ref VALID_NAME: regex::Regex = regex::Regex::new(r"^[a-zA-Z][a-
zA-Z0-9_]*$").unwrap();
714 +     }
715 +     let dbname = dbname.trim();
716 +     if dbname.is_empty() {
717 +         return Err(error::Error::BadRequest(
718 +             "Database name cannot be empty".to_string(),
719 +         ));
720 +     }
721 +     // PostgreSQL identifier limit is 63 bytes
722 +     if dbname.len() > 63 {
723 +         return Err(error::Error::BadRequest(
724 +             "Database name cannot exceed 63 characters".to_string(),
725 +         ));
713 726     }
714 727     if !VALID_NAME.is_match(dbname) {
715 728         return Err(error::Error::BadRequest(
716 -             "Catalog name must be alphanumeric, underscores
allowed".to_string(),
729 +             "Database name must start with a letter and contain only
alphanumeric characters or underscores".to_string(),
730 +         ));
731 +     }
732 +     // Additional check: block PostgreSQL reserved/special names

```

```
733 + let lower = dbname.to_lowercase();
734 + if lower == "template0" || lower == "template1" || lower == "postgres" {
735 +     return Err(error::Error::BadRequest(
736 +         "Cannot use reserved PostgreSQL database names".to_string(),
717 737         ));
718 738     }
719 739     if wmill_pg_creds.dbname.trim().eq_ignore_ascii_case(dbname.trim()) {
```



Comments 0



Please [sign in](#) to comment.