Software Design Specification (SDS)

Project Name: Shenron
Prepared By: Yusuf Tamer, Omar Yasser, Amr Khalid, Abdelrahman Alasil
Date: 11/30/2024

---

1. Introduction

1.1 Purpose

The purpose of this document is to describe the design, architecture, and technical specifications of Shenron. It outlines the functionality, system components, and design decisions to be followed during the development process.

1.2 Scope

This SDS covers the design and implementation details of the Shenron. The software will perform the following major tasks:

- Allow users to search for movies/series/anime.
- Allow users to view the details of a movie/series/anime.
- Allow users to rate a movie/series/anime.
- Allow users to post their opinion on a movie/series/anime through forums.
- Allow users to report other posts they found offensive or discriminatory.
- Allow moderators to remove reported posts from forums.
- Allow users to create a watch list and their favourite movies/series/animes to it.

---

2. System Overview

The system consists of the following components:

- **Frontend**: Html, CSS, and javascript
- **Backend**: Django version 5.1 , python version 3.11 or higher
- **Database**: MySQL 8.0 or higher

---

3. System Architecture

3.1 Architectural Design

This project follows the client-server architecture, where:

- **Frontend** communicates with the backend using RestApi.
- **Backend** interacts with the database to manage and retrieve data.

3.2 Data Flow

1. **User Interaction**: The user interacts with the UI to perform an action (e.g., search for a movie).
2. **Request Processing**: The frontend sends an API request to the backend server.
3. **Data Handling**: The backend processes the request, interacts with the database, and fetches or updates the necessary data.
4. **Response**: The backend sends the response back to the frontend, updating the UI.

---

4. Database Design

4.1 Database Schema

The system will store data in a relational MySQL database with the following entities and relationships:

1. **favourites_list_favourites**
   - **id**
   - **user_id**

2. **favourites_list_favourites_movies**
   - **id**
   - **favourites_id**
   - **movie_id**

3. **forum_comment**
   - **id**
   - **content**
   - **created_at**
   - **author_id**
   - **post_id**

4. **forum_post**
   - **id**
   - **content**
   - **title**
   - **created_at**
   - **author_id**
   - **topic_id**

5. **forum_topic**
   - **id**
   - **name**
   - **created_by**

6. **moviepage_movie**
   - **title**
   - **vote_average**
   - **vote_count**
   - **status**
   - **release_date**
   - **revenue**
   - **runtime**
   - **adult**
   - **backdrop_path**
   - **budget**
   - **original_language**
   - **overview**
   - **poster_path**
   - **tagline**
   - **genres**
   - **keywords**

7. user_user

- **id**
- **password**
- **last_login**
- **is_superuser**
- **username**
- **first_name**
- **last_name**
- **email**
- **is_staff**
- **is_active**
- **date_joined**

8. user_user_groups

- **id**
- **user_id**
- **group_id**

9. use_user_user_permissions

- **id**
- **user_id**
- **permission_id**

10. actor

- **actor_id**
- **name**
- **age**
- **total_programs**

11. actor_program

- **actor_id**
- **program_id**

12. use_user_user_permissions

- **post_id**
- **report_id**
- **user_id**
- **created_at**

4.2 Relationships:

1. **user_list (one-to-one):** a user can have a single list and a list can be owned by only one user.
2. **user_review (one-to-many):** A single user can make more than one review.
3. **forum_post (one-to-many):** each forum can contain multiple posts.
4. **programs_review (one-to-many):** each program can have multiple reviews.
5. **user_report (one-to-many):** each user can make a report or more.

6. **moderator_report (one-to-many):** each moderator will have a list of multiple reports assigned to them.
7. **program_actor (many-to-many):** every program contains multiple actors and every actor can act in multiple programs.
8. **user_post (one-to-many):** user can post multiple posts.
9. **program_forum (one-to-one):** each program can have one forum.
10. **list_program (many-to-many):** each program may appear in multiple lists and each list may have multiple programs.

---

5. Technology Stack

- **Frontend**: Javascript, HTML, and css.
- **Backend**: Django 5.1.
- **Database**: MySQL 8.0 or higher.
- **Hosting**: Verpex Managed Linux server-D4.

---

6. Testing Plan

6.1 Unit Testing

Each module and function will undergo unit testing to ensure that individual components are working as expected.

6.2 Integration Testing

Integration tests will validate that different modules (frontend and backend, or backend and database) work together as expected.

6.3 User Acceptance Testing (UAT)

End users will be involved in testing the system to verify that it meets their requirements and expectations.

6.4 Performance Testing

Stress and load testing will be conducted to ensure the system can handle the required number of users and operations without degradation in performance.

---

7. Conclusion

The Shenron is designed to fulfill the specified functional and non-functional requirements as described in this SDS. The design outlined here will ensure that the system is robust, scalable, and user-friendly, providing the intended value to its users.