**Online Application Form**

To design and implement the web-enabled system for submitting online application forms to various university courses, we can follow a multi-tier architecture. This architecture will consist of the following components:

1. Presentation Layer (Frontend): This layer will handle the user interface and interaction. It will be responsible for displaying the application form, capturing user inputs, and providing feedback to the users.

2. Application Layer (Backend): This layer will handle the application logic and data processing. It will be responsible for validating user inputs, storing the application data in the database, and generating unique application IDs.

3. Data Layer (Database): This layer will store and manage the application data. It will contain tables to store course details, applicant information, and other relevant data.

Based on the requirement specifications, here's a high-level schema design for the database:

1. Course Table:

- CourseID (Primary Key)

- Title

- Year

- Other details

2. Administrator Table:

- UserID (Primary Key)

- Password

- Other administrator details

3. Applicant Table:

- ApplicantID (Primary Key)

- CourseID (Foreign Key referencing Course Table)

- Name

- EntranceExamMarks

- LastUniversityAttended

- PreviousClassMarks

- Email

- Phone

- Other applicant details

With this schema design, you can implement the web-enabled system using a programming language like Python and a web framework like Django or Flask. Here's an overview of the implementation steps:

1. Set up the backend server using a web framework like Django or Flask.

2. Implement user management functionality, including authentication and authorization, for the administrator.

3. Create database models using an ORM (Object-Relational Mapping) tool provided by the chosen web framework.

4. Implement the necessary views and templates for the administrator to create and manage courses, view submitted applications, and perform other administrative tasks.

5. Implement the frontend interface for the applicants to fill out the application form and submit it.

6. Implement form validation on the server-side to ensure the correctness and completeness of the applicant's inputs.

7. Store the submitted application data in the database, generate a unique application ID, and notify the applicant of successful submission.

8. Provide appropriate feedback messages to the users throughout the application process.

Remember to follow security best practices, such as hashing and salting passwords, to protect sensitive user data. Additionally, consider implementing data validation, error handling, and other necessary features to enhance the usability and reliability of the system.

This is a high-level overview, and the actual implementation details may vary depending on your chosen technologies and specific requirements.