Exception handling:

Building an Employee Travel Approval Application for Corporates involves several components, including a user interface, database management, and integration of the approval workflow. Exception handling ensures that the application can manage unexpected errors or exceptional conditions that may arise during its execution. Here's a simplified example using Python:

```python
class EmployeeTravelApproval:

    def __init__(self):
        # Initialize necessary components
        pass


    def submit_request(self, employee_id, destination, travel_purpose):
        try:
            # Connect to the database and store the travel request
            pass
        except ConnectionError as e:
            print(f"Error: Failed to connect to the database. {e}")
        except Exception as e:
            print(f"An unexpected error occurred: {e}")


    def approve_request(self, request_id, approver_id):
        try:
            # Retrieve the request from the database and update its status
            pass
        except KeyError as e:
            print(f"Error: Request with ID {request_id} not found. {e}")
        except Exception as e:
            print(f"An unexpected error occurred: {e}")
```

```python
    def reject_request(self, request_id, rejecter_id, reason):
        try:
            # Retrieve the request from the database and update its status
            pass
        except KeyError as e:
            print(f"Error: Request with ID {request_id} not found. {e}")
        except Exception as e:
            print(f"An unexpected error occurred: {e}")


# Example usage
app = EmployeeTravelApproval()
app.submit_request("12345", "New York", "Business Meeting")
app.approve_request("TR123", "6789")
app.reject_request("TR456", "5555", "Budget constraints")
```